Project Report

On

**“GYM Management System”**

**Submitted for the partial fulfillment of the requirements of the BSC-IT 6th semester**



Submitted By : Debojit Gogoi

Under the Guidance of

Mr. Rajiv Kalita (Assistant Professor)

Department of Computer Science

*ACKNOWLEDGEMENT*

*We get immense pleasure to owe our sincere gratitude to all who helped us in doing this project. We especially thank the ‘WampServer’ & the ‘Navicat for MySQL Software’s,’ ‘html’. Without the help of these software perhaps this project would have remained failure.*

*We also thank our H.O.D Sir for his keen and consistent guidance, which has served as a beacon in the preparation of this project.*

*Next our heartfelt gratitude is to our parents for their unfailing support and encouragement in our every step.*

*And then at last but not the least, we acknowledge our being indebted friends without whose help and assistance this project would not have seen the light of the day.*

*Certificate*

*This is to certify that Debojit Gogoi of BSC-IT 6th semester from Jorhat Institute of Science and Technology, Jorhat have successfully completed this project entitled” Gym Management System” under my supervision and guidance as part of the assignment to be completed under the BSC-IT 6th semester course*

*Date: Internal Examiner*

*Place: Jorhat* *Department of Computer Science*

*Jorhat Institute of Science*

*and Technology, Jorhat*

**TABLE OF CONTENTS**

**CHAPTER 1: INTRODUCTION**

1.1 Background

1.2 Objective

1.3 Purpose, Scope and Applicability

1.3.1 Purpose

1.3.2 Scope

1.4 Achievements

1.5 Organization of Report

**CHAPTER 2: SURVEY OF TECHNOLOGY**

**CHAPTER 3: REQUIREMENTS AND ANALYSIS**

3.1 Problem Definition

3.2 Requirement Specification

3.3 Software and Hardware Requirements

3.3.1 Hardware Requirements

3.3.2 Software Requirements

3.4 Preliminary Product Description

3.5 Conceptual Models

3.6.1 Module Specifications

3.6.2 Screen Views

3.6.3 Data Flow Diagrams

**CHAPTER 4: SYSTEM DESIGN**

4.1 Basic Modules

4.2 Data Design

4.2.1 Schema Design

4.2.2 Data Integrity and Constraints

4.3 Procedural Design

4.3.1 Logic Diagrams

4.4 User Interface Design

4.5 Security Issues

**CHAPTER 5: IMPLEMENTATION AND TESTING**

5.1 Implementation Approaches

5.2 Coding Details and Code Efficiency

5.2.1 Coding

5.2.2 Code Efficiency

5.3 Testing Approach

5.3.1 Unit Testing

5.3.2 Integrated Testing

5.4 Modifications and Improvements

**CHAPTER 6: RESULTS AND DISCUSSION**

6.1 Test Reports

6.2 User Documentation

**CHAPTER 7: CONCLUSION**

7.1 Conclusion

7.2 Limitations of the System

7.3 Future scope of the Project

**CHAPTER 8: BIBLIOGRAPHY**

**Introduction:**

**GYM Management System** is the process whereby records of members are directly stored and can be accessed by the admin interactively in real-time without an intermediary service.

Gym Management System is the process of storing member’s details including health status, payment records, exercise routines etc. who have taken admission to the gym. Since the emergence of the World Wide Web, owners have sought to stored their user details in a digital system for easy access and find out every detail when needed.

**1.1 Background**

Gym Management System allows us to browse through endless possibilities, and access their customer’s details.

Say 'goodbye' to the days when you use to maintain a register, log book and bill. This system can store the payment history, generate bill, and store the details of the customers who have taken admission into the gym and also the exercise routine the respective member is following from his/her joining date. Payment records can be added and stored. Various plans of payments can be added with respective validity. The member enroll to the specific plan are displayed and their respective date of ending of their plan is also accessible which is fully controlled by the gym owner or the administrator.

* 1. **Objective**

My objective is to design such an application using which one can say 'goodbye' to the days when you store and maintain records of the users who have taken admission into the gym. Each and every records of the users are stored in the digital system which is accessible to the administrator and view the needed contents which saves time and convenient to use the system.

**1.3 Purpose and Scope**

1.3.1 Purpose

**GYM Management system** would have the following goals.

* Provide a web admin interface to add, view, delete records of all the customers.
* Provide an admin interface to view the total members of the gym.
* Provide an admin interface to change details of all the members when required.
* Provide an admin interface to view the total income from the members who have taken admission.

1.3.2 Scope

The main scope and deliverables of the project would be to:

* Understand and prepare detailed requirement and specifications
* Prepare high level and detailed design specifications of the system
* Prepare Test Plan and Test cases
* Develop the system and coding
* Perform unit testing, integration and system testing
* Demonstrate a bug free application after suitable modification if needed.

1.4 Achievements

* By successfully implementing the project, a substantial knowledge has been acquired on the implementation of a database system using .net technologies. This knowledge will be useful in the future in creating any type of desktop application or online database systems.

**2. SURVEY OF TECHNOLOGY**

In a desktop application like Gym Management System, there is a scope for a large number of platforms, languages and frameworks to choose from. Before selecting from this large array of technologies, the following aspects, which are characteristic to windows based application like this one, have been kept in mind:

* Data validation
* Performance
* Reliability
* Scalability
* Security
* Portability
* Performance
* Time constraint
* Cost constraint

The various technologies available for consideration are as follows:

**Operating System: Windows 7**

**Client Side Scripting:**

* **HTML**
* **CSS**
* **JavaScript**

**Server Side Scripting: PHP**

**Database Tool: My SQL**

**Testing Server: Apache**

**Other Software Used:**

* + - * **Adobe Dreamweaver**
      * **Adobe Photoshop**
      * **WampServer**

**HTML**

**HTML** or **Hypertext Markup Language** is the standard markup language used to create web pages.

HTML is written in the form of HTML elements consisting of *tags* enclosed in angle brackets (like <html>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent *empty elements* and so are unpaired, for example <img>. The first tag in a pair is the *start tag*, and the second tag is the *end tag* (they are also called *opening tags* and *closing tags*).

The purpose of a web browser is to read HTML documents and compose them into visible or audible web pages. The browser does not display the HTML tags, but uses the tags to interpret the content of the page. HTML describes the structure of a website semantically along with cues for presentation, making it a markup language rather than a programming language.

HTML elements form the building blocks of all websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML web pages.

**CSS**

CSS was first developed in 1997, as a way for Web developers to define the look and feel of their Web pages. It was intended to allow developers to separate content from design so that HTML could perform more of the function that it was originally based on the markup of content, without worry about the design and layout.

CSS didn't gain in popularity until around 2000, when Web browsers began using more than the basic font and color aspects of CSS.

Web Designers that don't use CSS for their design and development of Web sites are rapidly becoming a thing of the past. And it is arguably as important to understand CSS as it is to know HTML - and some would say it was more important to know CSS.

Style sheet refers to the document itself. Style sheets have been used for document design for years. They are the technical specifications for a layout, whether print or online. Print designers use style sheets to insure that their designs are printed exactly to specifications. A style sheet for a Web page serves the same purpose, but with the added functionality of also telling the viewing engine (the Web browser) how to render the document being viewed.

**PHP:**

PHP (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

Instead of lots of commands to output HTML (as seen in C or Perl), PHP pages contain HTML with embedded code that does "something" (in this case, output "Hi, I'm a PHP script!"). The PHP code is enclosed in special start and end processing instructions <?php and ?> that allow you to jump into and out of "PHP mode."

What distinguishes PHP from something like client-side JavaScript is that the code is executed on the server, generating HTML which is then sent to the client. The client would receive the results of running that script, but would not know what the underlying code was. You can even configure your web server to process all your HTML files with PHP, and then there's really no way that users can tell what you have up your sleeve.

The best things in using PHP are that it is extremely simple for a newcomer, but offers many advanced features for a professional programmer. Don't be afraid reading the long list of PHP's features. You can jump in, in a short time, and start writing simple scripts in a few hours.

**MYSQL:**

MySQL, the most popular Open Source SQL database management system, is developed, distributed, and supported by Oracle Corporation.

The MySQL Web site (http://www.mysql.com/) provides the latest information about MySQL software.

* **MySQL is a database management system.**

A database is a structured collection of data. It may be anything from a simple shopping list to a picture gallery or the vast amounts of information in a corporate network. To add, access, and process data stored in a computer database, you need a database management system such as MySQL Server. Since computers are very good at handling large amounts of data, database management systems play a central role in computing, as standalone utilities, or as parts of other applications.

* **MySQL databases are relational.**

A relational database stores data in separate tables rather than putting all the data in one big storeroom. The database structures are organized into physical files optimized for speed. The logical model, with objects such as databases, tables, views, rows, and columns, offers a flexible programming environment. You set up rules governing the relationships between different data fields, such as one-to-one, one-to-many, unique, required or optional, and “pointers” between different tables. The database enforces these rules, so that with a well-designed database, your application never sees inconsistent, duplicate, orphan, out-of-date, or missing data.

The SQL part of “MySQL” stands for “Structured Query Language”. SQL is the most common standardized language used to access databases. Depending on your programming environment, you might enter SQL directly (for example, to generate reports), embed SQL statements into code written in another language, or use a language-specific API that hides the SQL syntax.

SQL is defined by the ANSI/ISO SQL Standard. The SQL standard has been evolving since 1986 and several versions exist. In this manual, “SQL-92” refers to the standard released in 1992, “SQL:1999” refers to the standard released in 1999, and “SQL:2003” refers to the current version of the standard. We use the phrase “the SQL standard” to mean the current version of the SQL Standard at any time.

* **MySQL software is Open Source.**

Open Source means that it is possible for anyone to use and modify the software. Anybody can download the MySQL software from the Internet and use it without paying anything. If you wish, you may study the source code and change it to suit your needs. The MySQL software uses the GPL (GNU General Public License), http://www.fsf.org/licenses/, to define what you may and may not do with the software in different situations. If you feel uncomfortable with the GPL or need to embed MySQL code into a commercial application, you can buy a commercially licensed version from us. See the MySQL Licensing Overview for more information (http://www.mysql.com/company/legal/licensing/).

**3. REQUIREMENTS AND ANALYSIS**

**3.1 Problem Definition**

**Problem Definition and Need for the New System**

* Gym management System is a specific requirement of the client that integrates the storing and accessing services specifically for their members.
* Reports can be generated at any time within few seconds, so that manual labor is not required, and also analysis can be performed much more frequently which helps in taking decision.
* The details regarding all users, can also be maintained as their information is very helpful and sometimes becomes a critical requirement.
* To overcome these problems we develop “**GYM Management System”.**

**SYSTEM REQUIREMENTS SPECIFICATIONS**

System requirements are expressed in a software requirement document. The Software requirement specification (SRS) is the official statement of what is required of the system developers. This requirement document includes the requirements definition and the requirement specification. The software requirement document is not a design document. It should set out what the system should do without specifying how it should be done. The requirement set out in this document is complete and consistent.

The software specification document satisfies the following:-

* It specifies the external system behaviours.
* It specifies constraints on the implementation.
* It is easy to change.
* It serves as reference tool for system maintainers.
* It record forethought about the life cycle of the system.
* It characterizes acceptable response to undesired events.

**User Class and Characteristics:**

* Administrators can add, edit & delete member, plan, schedule and make payment update of the respective member.
* Administrator can view the monthly income of the member.
* Administrator can update the health status of the members.
* Administrator will have a overview of the total plan available total member join per year and month.
* Administrator can change admin password admin user profile and the secure key when the changes are required.
* Administrator can view the expire date of a specific member enrolled to the particular plan.

**Functional Requirements:**

* The System must provide following functionalities—
* Keeping records of registration of members.
* Keeping the records of payments.
* Keeping the monthly income.
* Storing the health status of the customer.

**Non Functional Requirements:**

Following Non-functional requirements will be there in the online shopping portal.

* Secure access of confidential data (customer’s details).
* 24 X 7 availability.
* Better component design to get better performance at peak time.

Flexible service based architecture will be highly desirable for future extension Nonfunctional requirements define system properties and constraints It arise through user needs, because of budget constraints or organizational policies, or due to the external factors such as safety regulations, privacy registration and so on.

Various other Non-functional requirements are:

1. Security

2. Reliability

3. Maintainability

4. Portability

5. Extensibility

6. Reusability

7. Application Affinity/Compatibility

8. Resource Utilization

**External Interface Requirements:**

**User Interface:**

User of the system will be provided with the Graphical user interface, there is no command line interface for any functions of the product.

**Hardware Interface**:

Hardware requirements for running this project are as follows:

Processor: - Pentium I or above.

RAM: - 128 MB or above.

HD: - 20 GB or above.

**Software Interface**:-

Software required to make working of product is:-

Front end- HTML/PHP

Back end- My SQL

**3.5 Conceptual Models**

**DATA FLOW DIAGRAM**

**What it is?**

The Data Flow Diagram shows the flow of data or information. It can be partitioned into single processes or functions. Data Flow Diagrams can be grouped together or decomposed into multiple processes. There can be physical DFD's that represent the physical files and transactions, or they can be business DFD's (logical, or conceptual).

**blankWhen it's used?**

The DFD is an excellent communication tool for analysts to model processes and functional requirements. One of the primary tools of the structured analysis efforts of the 1970's it was developed and enhanced by the likes of Yourdon, McMenamin, Palmer, Gane and Sarsen. It is still considered one of the best modeling techniques for eliciting and representing the processing requirements of a system.

Used effectively, it is a useful and easy to understand modeling tool. It has broad application and usability across most software development projects. It is easily integrated with data modeling, workflow modeling tools, and textual specs. Together with these, it provides analysts and developers with solid models and specs. Alone, however, it has limited usability. It is simple and easy to understand by users and can be easily extended and refined with further specification into a physical version for the design and development teams.

The different versions are Context Diagrams (Level 0), Partitioned Diagrams (single process only -- one level), functionally decomposed, leveled sets of Data Flow Diagrams.

**Data Store**

It is a repository of information. In the physical model, this represents a file, table, etc. In the logical model, a data store is an object or entity.

**Dataflow**   
 DFDs show the flow of data from external entities into the system, showed how the data moved from one process to another, as well as its logical storage. There are only four symbols:

* Squares representing **external entities**, which are sources or destinations of data.
* Rounded rectangles representing **processes**, which take data as input, do something to it, and output it.
* Arrows representing the **data flows**, which can either, be electronic data or physical items.
* Open-ended rectangles representing **data stores**, including electronic stores such as databases or XML files and physical stores such as or filing cabinets or stacks of paper.

There are several common modeling rules for creating DFDs:

* All processes must have at least one data flow in and one data flow out.
* All processes should modify the incoming data, producing new forms of outgoing data.
* Each data store must be involved with at least one data flow.
* Each external entity must be involved with at least one data flow.
* A data flow must be attached to at least one process.

DFDs are nothing more than a network of related system functions and indicate from where information is received and to where it is sent. It is the starting point in the system that decomposes the requirement specifications down to the lowest level detail.

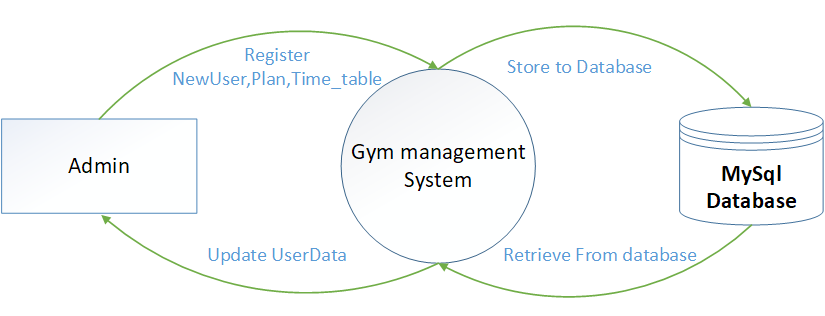
The four symbols in DFD, each of which has its meaning. They are given below:

* External entities are outside to system but they either supply input data in the system or use the system output. These are represented by square of rectangle. External entities that supply data into a system are sometimes called Sources. External entities that use system data are sometimes called sinks.
* Dataflow models that passages of data in the system and are represented by line by joining system components. An arrow indicates the direction of the flow and the line is labeled by the name of the dataflow.
* Process show that the systems do. Each process has one or more data inputs and one or data outputs. Circles in DFD represent them. Each high level process may be consisting of more than one lower level processes. Process will be expanded in sequent level DFD. A circle or a bubble represents a process that transforms incoming data flow into outgoing dataflow.

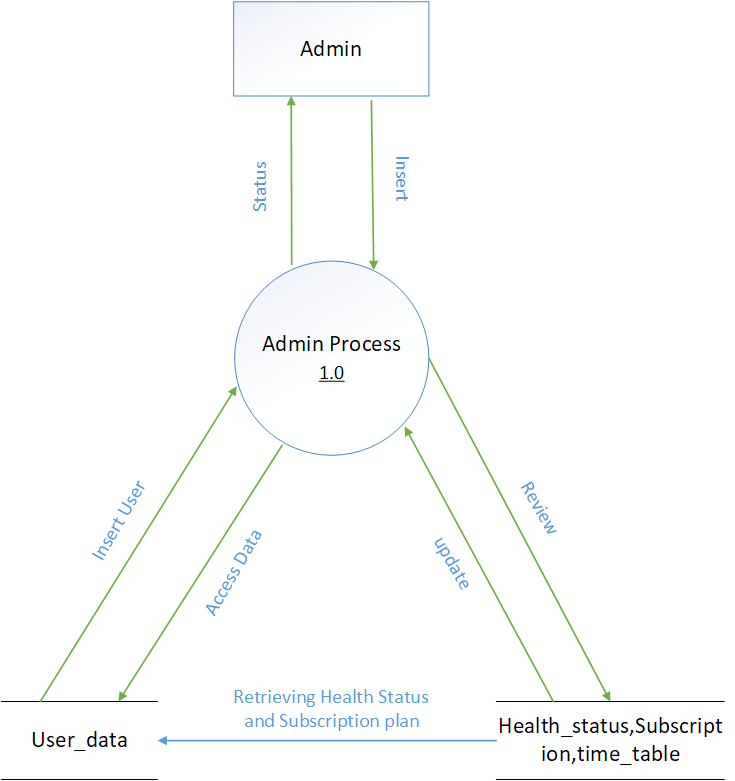
The high level processes in a system are:

* Receivable process.
* Verifiable process.
* Disposal process.
* File or data store is a repository of data. They contain data that is retained in the system. Process can enter data into data store or retrieved data from the data store. An open rectangle is a data store, data at rest.

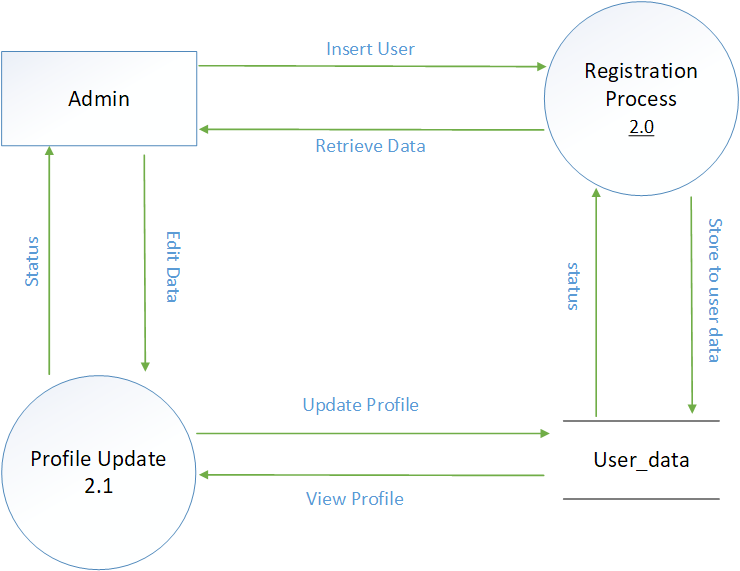
**0-Level DFD:**

****

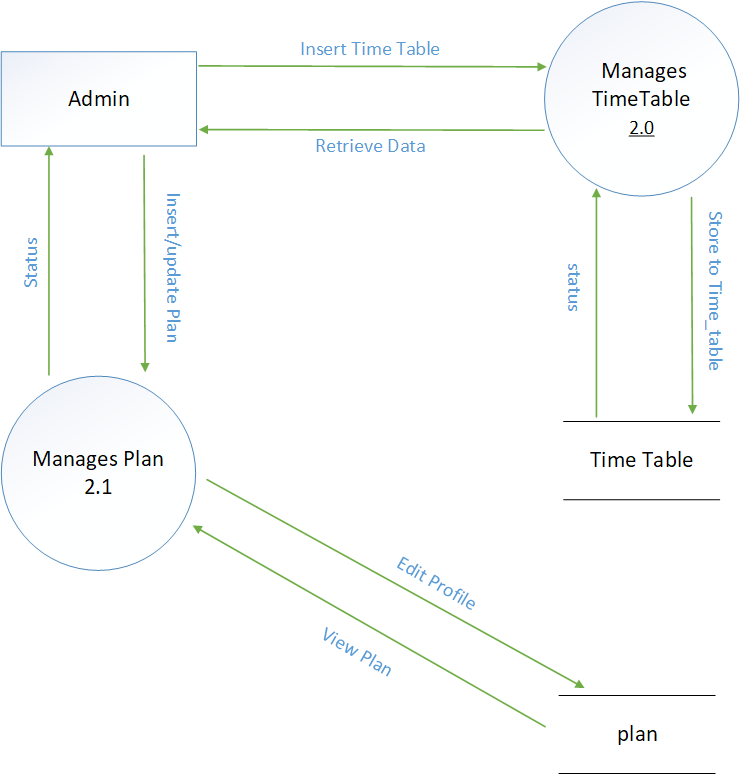
**1st level DFD for Admin Processes**

****

**DFD For User Registration and Profile Update**

****

**DFD for Manage Timetable/Plan**

****

# Entity-Relationship Model

Simply stated the ER model is a conceptual data model that views the real world as entities and relationships. A basic component of the model is the Entity-Relationship diagram which is used to visually represent data objects. Since Chen wrote his paper the model has been extended and today it is commonly used for database.

## Basic Constructs of E-R Modeling

The ER model views the real world as a construct of entities and association between entities.

## Entities

Entitiesare the principal data object about which information is to be collected. Entities are classified as independent or dependent (in some methodologies, the terms used are strong and weak, respectively). An independent entityis one that does not rely on another for identification. A dependent entityis one that relies on another for identification. .

## Relationships

A Relationship represents an association between two or more entities. Relationships are classified in terms of degree, connectivity, cardinality, and existence.

## Attributes

Attributesdescribe the entity of which they are associated. A particular instance of an attribute is a value. The domain of an attribute is the collection of all possible values an attribute can have. The domain of Name is a character string.

## Classifying Relationships

Relationships are classified by their degree, connectivity, cardinality, direction, type, and existence. Not all modeling methodologies use all these classifications.

## Degree of a Relationship

The degree of a relationship is the number of entities associated with the relationship. The n-ary relationship is the general form for degree n. Special cases are the binary, and ternary, where the degree is 2 and 3 respectively.

**Connectivity and Cardinality**

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one" or "many". The cardinality of a relationship is the actual number of related occurrences for each of the two entities. The basic types of connectivity for relations are: one-to-one, one-to-many, and many-to-many.

## Direction

The direction of a relationship indicates the originating entity of a binary relationship. The entity from which a relationship originates is the parent entity; the entity where the relationship terminates is the child entity.

The direction of a relationship is determined by its connectivity type .An identifying relationshipis one in which one of the child entities is also a dependent entity. A non-identifying relationshipis one in which both entities are independent.

## Existence

Existencedenotes whether the existence of an entity instance is dependent upon the existence of another, related, entity instance. The existence of an entity in a relationship is defined as either mandatory or optional.

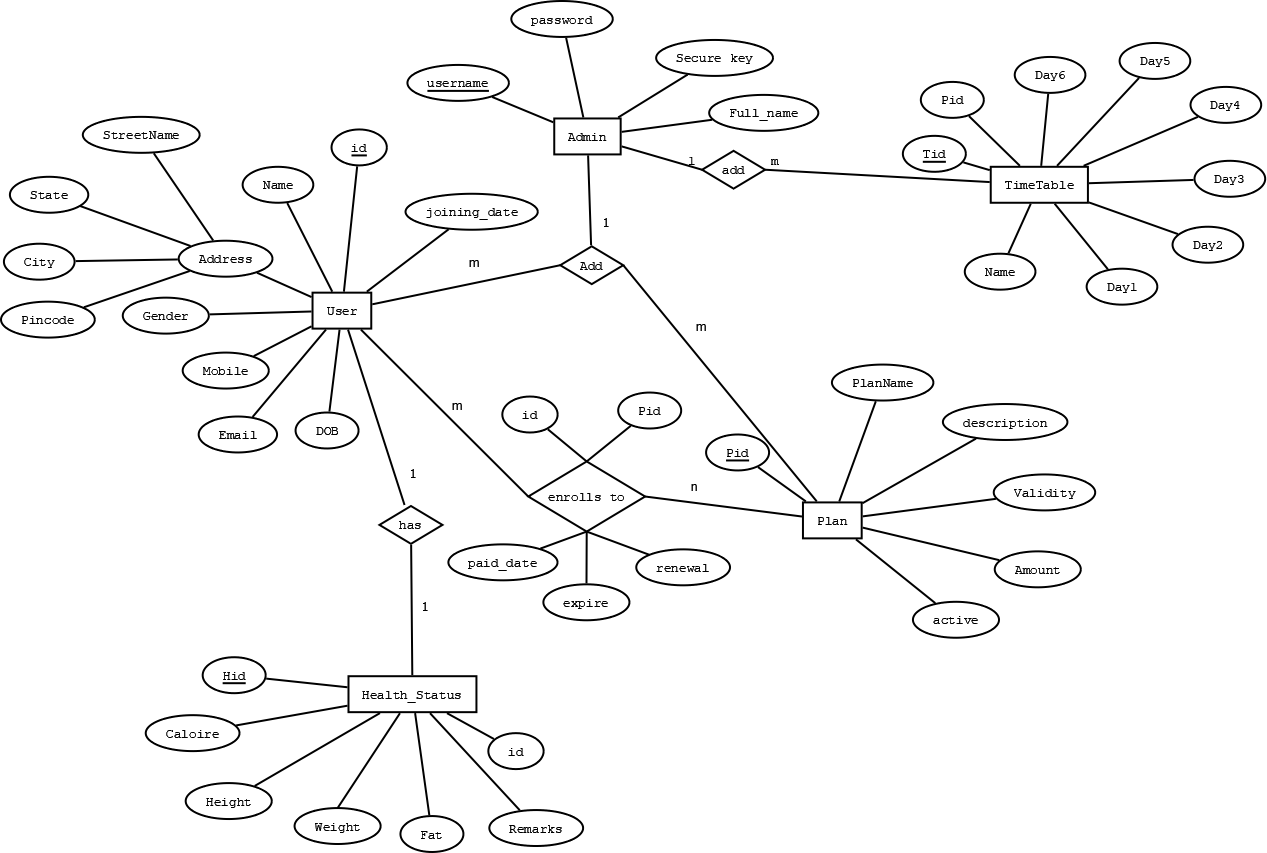
## Generalization Hierarchies

A generalization hierarchy is a form of abstraction that specifies that two or more entities that share common attributes can be generalized into a higher level entity type called a super typeor genericentity. The lower-level of entities become the subtype, or categories, to the super type. Subtypes are dependent entities.

**ER Notation**

The symbols used for the basic ER constructs are:

* Entities are represented by labeled rectangles. The label is the name of the entity.
* Relationships are represented by a solid line connecting two entities. The name of the relationship is written above the line. Relationship names should be verbs.
* Attributes, when included, are listed inside the entity rectangle. Attributes which are identifiers are underlined. Attribute names should be singular nouns.
* Cardinality of many is represented by a line ending in a crow's foot. If the crow's foot is omitted, the cardinality is one.
* Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.
* Existence is represented by placing a circle or a perpendicular bar on the line. Mandatory existence is shown by the bar (looks like a 1) next to the entity for an instance is required. Optional existence is shown by placing a circle next to the entity that is optional.

****

**ER Diagram**

1. **SYSTEM DESIGN**

**Introduction:**

System design is the solution of a “how to approach to the creation of the new system. It is composed of several steps. It facilitates the understanding and provides the procedural details necessary for implementation of the system recommended in the feasibility study. Emphasis is given on translating the performance requirements into design specification. Design goes through logical and physical stages of development.

Logical design reviews the present physical system; prepares input and output specification; make editing; security and control specification; details the implementation plan, and prepare logical design walk through. The physical design maps out the details of the physical system; plans the system implementation plan and specifies hardware and software. System design translates the system requirement into the ways of the system as recommended in the feasibility study. Thus the system design is the translation from user-oriented document to a programmer or a database personal oriented document. System design is a highly creative process that can be greatly facilitated by the following:-

* Strong Problem Definition
* Pictorial description of the Existing System
* Set of Requirements of the new system

**Modules Description**:

1. Registration: Customer can register their account here to continue shopping.
2. Admin: Admin can add books, check orders and make sure the orders are delivered on time and can confirm payments by the customers.
3. Shopping Cart: Customers after login can browse through the different books and choose one or more products and can add them to cart.
4. Payment: Cash on Delivery facility is available.

**3.2 INPUT DESIGN**

Very careful attention had to be given to input design, which is a major part of the overall system design. In order to make the data entry as easy, logical and error free as possible, specific standards had been followed. Validation checks, provided in the system prevented the user in entering incorrect, erroneous data. This made sure that, only valid data had been available for data processing. If valid data was entered, then meaningful error messages had been prompted to enter correct data. The interactive screen formats facilitate the entry of valid data.

**3.2.1 VALIDATIONS:**

Some fields are having only number, as an I/P. For this key ASCII is checked. If they entered characters, it would display the message to enter number only. Exchange rates field will be validated for number and dot symbols.

**3.2.2 INPUT DESIGN OBJECTIVES:**

The numbers of clear objectives of input design are,

* To produce a cost effective method of input
* To achieve the highest possible level of accuracy
* To ensure that the input is acceptable to and understand by the user staff

**3.3 OUTPUT DESIGN:**

Output, as you probably know, generally refers to the results and information that are generated by the system. For many end-users, output is the main reason for developing the system and the basis on which they will evaluate the usefulness of the application. Most end users will not actually operate the information system or enter data through workstations, but they will use the output from the system.

When designing output, systems analysts must accomplish the following.

* Determine what information to present
* Decide whether to display, print, or “speak” the information and select the output medium.
* Arrange the presentation of information in an acceptable format.
* Decide how to distribute the output to intended recipients.

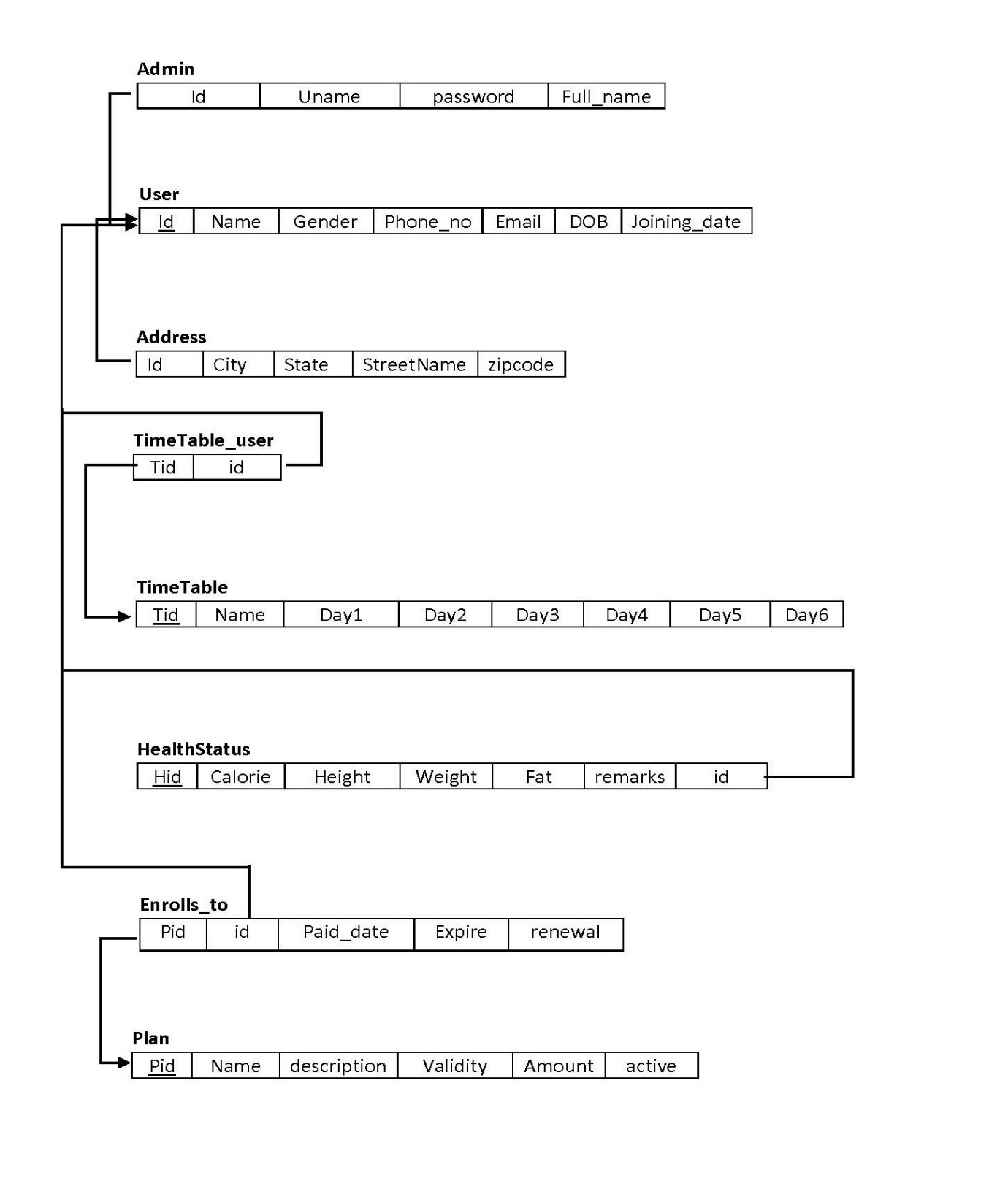
That alignment of information on a display or printed document is termed as layout.

Accomplishing the general activities listed above will require specific decisions, such as whether to use preprinted forms when preparing reports and documents, how many lines to plan on a printed page, or whether to use graphics and color.

The output design is specified on layout performs, sheets that describe the location characteristics, and format of the column headings and pagination. As we indicated at the beginning of this discussion, these elements are analogous to an architect’s blue print that shows the location of the each component.

**3.4 Relational Model**

Relational Model was proposed by E.F. Codd to model data in the form of relations or tables. After designing the conceptual model of Database using ER diagram, we need to convert the conceptual model in the relational model which can be implemented using any RDMBS languages like Oracle SQL, MySQL etc.

****

**5. IMPLEMENTATION AND TESTING**

**5.1 Implementation approaches**

The Software Design Description Document has been used as input in the implementation process. The actual implementation has been done using PHP. PHP has been used to interact with the backend database. In this implementation, My SQL Server has been used as the backend RDBMS. PHP processes the inputs or commands given by the user and translates them in the commands understandable to the backend database. The output produced by the backend database is also handled by PHP which then displayed on the Browser screen.

**Coding and Screenshots**

**Index Page**

**Registration Page**

**View Member**

**Manage plan Page**

**Coding of index.php Page**

<?php

require '../../include/db\_conn.php';

page\_protect();

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title>Titan Gym | Dashboard </title>

<link rel="stylesheet" href="../../css/style.css" id="style-resource-5">

<script type="text/javascript" src="../../js/Script.js"></script>

<link rel="stylesheet" href="../../css/dashMain.css">

<link rel="stylesheet" type="text/css" href="../../css/entypo.css">

<style>

.page-container .sidebar-menu #main-menu li#dash > a {

background-color: #2b303a;

color: #ffffff;

}

</style>

</head>

<body class="page-body page-fade" onload="collapseSidebar()">

<div class="page-container sidebar-collapsed" id="navbarcollapse">

<div class="sidebar-menu">

<header class="logo-env">

<!-- logo -->

<div class="logo">

<a href="main.php">

<img src="../../images/logo.png" alt="" width="192" height="80" /></a>

</div>

<!-- logo collapse icon -->

<div class="sidebar-collapse" onclick="collapseSidebar()">

<a href="#" class="sidebar-collapse-icon with-animation"><!-- add class "with-animation" if you want sidebar to have animation during expanding/collapsing transition -->

<i class="entypo-menu"></i>

</a>

</div>

</header>

<?php include('nav.php'); ?>

</div>

<div class="main-content">

<div class="row">

<!-- Profile Info and Notifications -->

<div class="col-md-6 col-sm-8 clearfix">

</div>

<!-- Raw Links -->

<div class="col-md-6 col-sm-4 clearfix hidden-xs">

<ul class="list-inline links-list pull-right">

<li>Welcome <?php echo $\_SESSION['full\_name']; ?>

</li>

<li>

<a href="logout.php">Log Out <i class="entypo-logout right"></i></a>

</li>

</ul>

</div>

</div>

<h2>Titan Gym</h2>

<hr>

<div class="col-sm-3"><a href="revenue\_month.php">

<div class="tile-stats tile-red">

<div class="icon"><i class="entypo-users"></i></div>

<div class="num" data-postfix="" data-duration="1500" data-delay="0">

<h2>Paid Income This Month</h2><br>

<?php

date\_default\_timezone\_set("Asia/Calcutta");

$date = date('Y-m');

$query = "select \* from enrolls\_to WHERE paid\_date LIKE '$date%'";

//echo $query;

$result = mysqli\_query($con, $query);

$revenue = 0;

if (mysqli\_affected\_rows($con) != 0) {

while ($row = mysqli\_fetch\_array($result, MYSQLI\_ASSOC)) {

$query1="select \* from plan where pid='".$row['pid']."'";

$result1=mysqli\_query($con,$query1);

if($result1){

$value=mysqli\_fetch\_row($result1);

$revenue = $value[4] + $revenue;

}

}

}

echo "₹".$revenue;

?>

</div>

</div></a>

</div>

<div class="col-sm-3"><a href="table\_view.php">

<div class="tile-stats tile-green">

<div class="icon"><i class="entypo-chart-bar"></i></div>

<div class="num" data-postfix="" data-duration="1500" data-delay="0">

<h2>Total <br>Members</h2><br>

<?php

$query = "select COUNT(\*) from users";

$result = mysqli\_query($con, $query);

$i = 1;

if (mysqli\_affected\_rows($con) != 0) {

while ($row = mysqli\_fetch\_array($result, MYSQLI\_ASSOC)) {

echo $row['COUNT(\*)'];

}

}

$i = 1;

?>

</div>

</div></a>

</div>

<div class="col-sm-3"><a href="over\_members\_month.php">

<div class="tile-stats tile-aqua">

<div class="icon"><i class="entypo-mail"></i></div>

<div class="num" data-postfix="" data-duration="1500" data-delay="0">

<h2>Joined This Month</h2><br>

<?php

date\_default\_timezone\_set("Asia/Calcutta");

$date = date('Y-m');

$query = "select COUNT(\*) from users WHERE joining\_date LIKE '$date%'";

$result = mysqli\_query($con, $query);

$i = 1;

if (mysqli\_affected\_rows($con) != 0) {

while ($row = mysqli\_fetch\_array($result, MYSQLI\_ASSOC)) {

echo $row['COUNT(\*)'];

}

}

$i = 1;

?>

</div>

</div></a>

</div>

<div class="col-sm-3"><a href="view\_plan.php">

<div class="tile-stats tile-blue">

<div class="icon"><i class="entypo-rss"></i></div>

<div class="num" data-postfix="" data-duration="1500" data-delay="0">

<h2>Total Plan Available</h2><br>

<?php

$query = "select COUNT(\*) from plan where active='yes'";

$result = mysqli\_query($con, $query);

$i = 1;

if (mysqli\_affected\_rows($con) != 0) {

while ($row = mysqli\_fetch\_array($result, MYSQLI\_ASSOC)) {

echo $row['COUNT(\*)']; }

}

$i = 1;

?>

</div>

</div></a>

</div>

<?php include('footer.php'); ?>

</div>

</body>

</html>

**new\_entry.php**

<?php

require '../../include/db\_conn.php';

page\_protect();

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title>Titan Gym | New User</title>

<link rel="stylesheet" href="../../css/style.css" id="style-resource-5">

<script type="text/javascript" src="../../js/Script.js"></script>

<link rel="stylesheet" href="../../css/dashMain.css">

<link rel="stylesheet" type="text/css" href="../../css/entypo.css">

<link href="a1style.css" type="text/css" rel="stylesheet">

<style>

.page-container .sidebar-menu #main-menu li#regis > a {

background-color: #2b303a;

color: #ffffff;

}

#boxx

{

width:220px;

}

</style>

</head>

<body class="page-body page-fade" onload="collapseSidebar()">

<div class="page-container sidebar-collapsed" id="navbarcollapse">

<div class="sidebar-menu">

<header class="logo-env">

<!-- logo -->

<div class="logo">

<a href="main.php">

<img src="../../images/logo.png" alt="" width="192" height="80" />

</a>

</div>

<!-- logo collapse icon -->

<div class="sidebar-collapse" onclick="collapseSidebar()">

<a href="#" class="sidebar-collapse-icon with-animation"><!-- add class "with-animation" if you want sidebar to have animation during expanding/collapsing transition -->

<i class="entypo-menu"></i>

</a>

</div>

</header>

<?php include('nav.php'); ?>

</div>

<div class="main-content">

<div class="row">

<!-- Profile Info and Notifications -->

<div class="col-md-6 col-sm-8 clearfix">

</div>

<!-- Raw Links -->

<div class="col-md-6 col-sm-4 clearfix hidden-xs">

<ul class="list-inline links-list pull-right">

<li>Welcome <?php echo $\_SESSION['full\_name']; ?>

</li>

<li>

<a href="logout.php">

Log Out <i class="entypo-logout right"></i>

</a>

</li>

</ul>

</div>

</div>

<h3>New Registration</h3>

<hr />

<div class="a1-container a1-small a1-padding-32" style="margin-top:2px; margin-bottom:2px;">

<div class="a1-card-8 a1-light-gray" style="width:500px; margin:0 auto;">

<div class="a1-container a1-dark-gray a1-center">

<h6>NEW ENTRY</h6>

</div>

<form id="form1" name="form1" method="post" class="a1-container" action="new\_submit.php">

<table width="100%" border="0" align="center">

<tr>

<td height="35"><table width="100%" border="0" align="center">

<tr>

<td height="35">MEMBERSHIP ID:</td>

<td height="35"><input type="text" id="boxx" name="m\_id" value="<?php echo time(); ?>" readonly required/></td>

</tr>

<tr>

<td height="35">NAME:</td>

<td height="35"><input name="u\_name" id="boxx" required/></td>

</tr>

<tr>

<td height="35">STREET NAME:</td>

<td height="35"><input name="street\_name" id="boxx" required/></td>

</tr>

<tr>

<td height="35">CITY:</td>

<td height="35"><input <input type="text" name="city" id="boxx" required/ ></td>

</tr>

<tr>

<td height="35">ZIPCODE:</td>

<td height="35"><input type="number" name="zipcode" id="boxx" maxlength="6" required / ></td>

</tr>

<tr>

<td height="35">STATE:</td>

<td height="35"><input type="text" name="state" id="boxx" required/ size="30"></td>

</tr>

<tr>

<td height="35">GENDER:</td>

<td height="35"><select name="gender" id="boxx" required>

<option value="">--Please Select--</option>

<option value="Male">Male</option>

<option value="Female">Female</option>

</select></td>

</tr>

<tr>

<td height="35">DATE OF BIRTH:</td>

<td height="35"><input type="date" name="dob" id="boxx" required/ size="30"></td>

</tr>

<tr>

<td height="35">PHONE NO:</td>

<td height="35"><input type="number" name="mobile" id="boxx" maxlength="10" required/ size="30"></td>

</tr>

<tr>

<td height="35">EMAIL ID:</td>

<td height="35"><input type="email" name="email" id="boxx" required/ size="30"></td>

</tr>

<tr>

<td height="35">JOINING DATE:</td>

<td height="35"><input type="date" name="jdate" id="boxx" required size="30"></td>

</tr>

<tr>

<td height="35">PLAN:</td>

<td height="35"><select name="plan" id="boxx" required onchange="myplandetail(this.value)">

<option value="">--Please Select--</option>

<?php

$query="select \* from plan where active='yes'";

$result=mysqli\_query($con,$query);

if(mysqli\_affected\_rows($con)!=0){

while($row=mysqli\_fetch\_row($result)){

echo "<option value=".$row[0].">".$row[1]."</option>";

}

}

?>

</select></td>

</tr>

<tbody id="plandetls">

</tbody>

<tr>

<tr>

<td height="35">&nbsp;</td>

<td height="35"><input class="a1-btn a1-blue" type="submit" name="submit" id="submit" value="Register" >

<input class="a1-btn a1-blue" type="reset" name="reset" id="reset" value="Reset"></td>

</tr>

</table></td>

</tr>

</table>

</form>

</div>

</div>

<script>

function myplandetail(str){

if(str==""){

document.getElementById("plandetls").innerHTML = "";

return;

}else{

if (window.XMLHttpRequest) {

// code for IE7+, Firefox, Chrome, Opera, Safari

xmlhttp = new XMLHttpRequest();

}

xmlhttp.onreadystatechange = function() {

if (this.readyState == 4 && this.status == 200) {

document.getElementById("plandetls").innerHTML=this.responseText;

}

};

xmlhttp.open("GET","plandetail.php?q="+str,true);

xmlhttp.send();

}

}

</script>

<?php include('footer.php'); ?>

</div>

</body>

</html>

**viewall\_detail.php**

<?php

require '../../include/db\_conn.php';

page\_protect();

if (isset($\_POST['name'])) {

$memid = $\_POST['name'];

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title>Titan Gym | Edit Member</title>

<link rel="stylesheet" href="../../css/style.css" id="style-resource-5">

<script type="text/javascript" src="../../js/Script.js"></script>

<link rel="stylesheet" href="../../css/dashMain.css">

<link rel="stylesheet" type="text/css" href="../../css/entypo.css">

<link href="a1style.css" rel="stylesheet" type="text/css">

<style>

#button1

{

width:126px;

}

#boxxe

{

width:230px;

}

.page-container .sidebar-menu #main-menu li#hassubopen > a {

background-color: #2b303a;

color: #ffffff;

}

</style>

</head>

<body class="page-body page-fade" onload="collapseSidebar()">

<div class="page-container sidebar-collapsed" id="navbarcollapse">

<div class="sidebar-menu">

<header class="logo-env">

<!-- logo -->

<div class="logo">

<a href="main.php">

<img src="../../images/logo.png" alt="" width="192" height="80" />

</a>

</div>

<!-- logo collapse icon -->

<div class="sidebar-collapse" onclick="collapseSidebar()">

<a href="#" class="sidebar-collapse-icon with-animation"><!-- add class "with-animation" if you want sidebar to have animation during expanding/collapsing transition -->

<i class="entypo-menu"></i>

</a>

</div>

</header>

<?php include('nav.php'); ?>

</div>

<div class="main-content">

<div class="row">

<!-- Profile Info and Notifications -->

<div class="col-md-6 col-sm-8 clearfix">

</div>

<!-- Raw Links -->

<div class="col-md-6 col-sm-4 clearfix hidden-xs">

<ul class="list-inline links-list pull-right">

<li>

Welcome <?php echo $\_SESSION['full\_name']; ?> </li>

<li>

<a href="logout.php">Log Out <i class="entypo-logout right"></i></a>

</li>

</ul>

</div>

</div>

<h3>Edit Member Details</h3>

<hr />

<?php

$query = "SELECT \* FROM users u INNER JOIN address a ON u.userid=a.id

INNER JOIN health\_status h ON u.userid=h.uid

INNER JOIN enrolls\_to e on u.userid=e.uid

INNER JOIN plan p on e.pid=p.pid

WHERE userid='$memid' and e.renewal='yes'";

$result = mysqli\_query($con, $query);

$sno = 1;

$name="";

$gender="";

if (mysqli\_affected\_rows($con) == 1) {

while ($row = mysqli\_fetch\_array($result, MYSQLI\_ASSOC)) {

$name = $row['username'];

$gender =$row['gender'];

$mobile = $row['mobile'];

$email = $row['email'];

$dob = $row['dob'];

$jdate = $row['joining\_date'];

$streetname=$row['streetName'];

$state=$row['state'];

$city=$row['city'];

$zipcode=$row['zipcode'];

$calorie=$row['calorie'];

$height=$row['height'];

$weight=$row['weight'];

$fat=$row['fat'];

$planname=$row['planName'];

$pamount=$row['amount'];

$pvalidity=$row['validity'];

$pdescription=$row['description'];

$paiddate=$row['paid\_date'];

$expire=$row['expire'];

$remarks=$row['remarks'];

}

}

else{

echo "<html><head><script>alert('Change Unsuccessful');</script></head></html>";

echo mysqli\_error($con);

}

?>

<div class="a1-container a1-small a1-padding-32" style="margin-top:2px; margin-bottom:2px;">

<div class="a1-card-8 a1-light-gray" style="width:600px; margin:0 auto;">

<div class="a1-container a1-dark-gray a1-center">

<h6>Edit Member Details</h3>

</div>

<form id="form1" name="form1" method="post" class="a1-container" action="edit\_member.php">

<table width="100%" border="0" align="center">

<tbody><tr>

<td height="35">

</td></tr><tr>

<td height="35">USER ID:</td>

<td height="35"><input type="text" name="name" id="boxxe" readonly="" required="" value='<?php echo $memid?>'></td>

</tr>

<tr>

<td height="35">NAME:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $name?>'></td>

</tr>

<tr>

<td height="35">GENDER:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $gender?>'></td>

</tr>

<tr>

<td height="35">MOBILE:</td>

<td height="35"><input type="text" id="boxxe" readonly="" maxlength="10" value='<?php echo $mobile ?>'></td>

</tr>

<tr>

<td height="35">EMAIL:</td>

<td height="35"><input type="email" id="boxxe" readonly="" required="" value='<?php echo $email?>'></td>

</tr>

<tr>

<td height="35">DATE OF BIRTH</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $dob?>'></td>

</tr>

<tr>

<td height="35">JOINING DATE:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $jdate?>'></td>

</tr>

<tr>

<td height="35">STREET NAME:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $streetname?>'></td>

</tr>

<tr>

<td height="35">STATE:</td>

<td height="35"><input type="text" id="boxxe" readonly="" name="state" value='<?php echo $state?>'></td>

</tr>

<tr>

<td height="35">CITY:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $city?>'></td>

</tr>

<tr>

<td height="35">ZIPCODE:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $zipcode?>'></td>

</tr>

<tr>

<td height="35">CALORIE:</td>

<td height="35"><input type="text" id="boxxe" readonly="" value='<?php echo $calorie?>'></td>

</tr>

<tr>

<td height="35">HEIGHT:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $height?>'></td>

</tr>

<tr>

<td height="35">WEIGHT:</td>

<td height="35"><input type="text" readonly="" value='<?php echo $weight?>' id="boxxe"></td>

</tr>

<tr>

<td height="35">FAT:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $weight?>'></td>

</tr>

<tr>

<td height="35">PLAN NAME:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $planname?>'></td>

</tr>

<tr>

<td height="35">PLAN AMOUNT:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $pamount?>'></td>

</tr>

<tr>

<td height="35">PLAN VALIDITY:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $pvalidity.' Month'?>'></td>

</tr>

<tr>

<td height="35">PLAN DESCRIPTION:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $pdescription?>'></td>

</tr>

<tr>

<td height="35">PAID DATE:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $paiddate?>'></td>

</tr>

<tr>

<td height="35">EXPIRED DATE:</td>

<td height="35"><input type="text" readonly="" id="boxxe" value='<?php echo $expire?>'></td>

</tr>

<tr>

<td height="35">REMARKS:</td>

<td height="35"><textarea readonly style="resize:none; margin: 0px; width: 230px; height: 53px;" ><?php echo $remarks?></textarea></td>

</tr>

<tr>

</tr><tr>

<td height="35">&nbsp;</td>

<td height="35"><input class="a1-btn a1-blue" type="submit" name="submit" id="submit" value="EDIT">

<a href="table\_view"><input class="a1-btn a1-blue" id="" value="BACK"></a></td>

</tr>

</tbody></table>

</div>

</div>

<?php include('footer.php'); ?>

</div>

</body>

</html>

<?php

} else {

}

?>

**view\_mem.php**

<?php

require '../../include/db\_conn.php';

page\_protect();

?>

<!DOCTYPE html>

<html lang="en">

<head>

<title>Titan Gym | Member View</title>

<link rel="stylesheet" href="../../css/style.css" id="style-resource-5">

<script type="text/javascript" src="../../js/Script.js"></script>

<link rel="stylesheet" href="../../css/dashMain.css">

<link rel="stylesheet" type="text/css" href="../../css/entypo.css">

<link href="a1style.css" rel="stylesheet" type="text/css">

<style>

#button1

{

width:126px;

}

.page-container .sidebar-menu #main-menu li#hassubopen > a {

background-color: #2b303a;

color: #ffffff;

}

</style>

</head>

<body class="page-body page-fade" onload="collapseSidebar()">

<div class="page-container sidebar-collapsed" id="navbarcollapse">

<div class="sidebar-menu">

<header class="logo-env">

<!-- logo -->

<div class="logo">

<a href="main.php">

<img src="../../images/logo.png" alt="" width="192" height="80" />

</a>

</div>

<!-- logo collapse icon -->

<div class="sidebar-collapse" onclick="collapseSidebar()">

<a href="#" class="sidebar-collapse-icon with-animation"><!-- add class "with-animation" if you want sidebar to have animation during expanding/collapsing transition -->

<i class="entypo-menu"></i>

</a>

</div>

</header>

<?php include('nav.php'); ?>

</div>

<div class="main-content">

<div class="row">

<!-- Profile Info and Notifications -->

<div class="col-md-6 col-sm-8 clearfix">

</div>

<!-- Raw Links -->

<div class="col-md-6 col-sm-4 clearfix hidden-xs">

<ul class="list-inline links-list pull-right">

<li>Welcome <?php echo $\_SESSION['full\_name']; ?>

</li>

<li>

<a href="logout.php">

Log Out <i class="entypo-logout right"></i>

</a>

</li>

</ul>

</div>

</div>

<h3>Edit Member</h3>

<hr />

<table class="table table-bordered datatable" id="table-1" border=1>

<thead>

<tr><h2>

<th>Sl.No</th>

<th>Membership Expiry</th>

<th>Member ID</th>

<th>Name</th>

<th>Contact</th>

<th>E-Mail</th>

<th>Gender</th>

<th>Joining Date</th>

<th>Action</th></h2>

</tr>

</thead>

<tbody>

<?php

$query = "select \* from users ORDER BY joining\_date";

//echo $query;

$result = mysqli\_query($con, $query);

$sno = 1;

if (mysqli\_affected\_rows($con) != 0) {

while ($row = mysqli\_fetch\_array($result, MYSQLI\_ASSOC)) {

$uid = $row['userid'];

$query1 = "select \* from enrolls\_to WHERE uid='$uid' AND renewal='yes'";

$result1 = mysqli\_query($con, $query1);

if (mysqli\_affected\_rows($con) == 1) {

while ($row1 = mysqli\_fetch\_array($result1, MYSQLI\_ASSOC)) {

echo "<tr><td>".$sno."</td>";

echo "<td>" . $row1['expire'] . "</td>";

echo "<td>" . $row['userid'] . "</td>";

echo "<td>" . $row['username'] . "</td>";

echo "<td>" . $row['mobile'] . "</td>";

echo "<td>" . $row['email'] . "</td>";

echo "<td>" . $row['gender'] . "</td>";

echo "<td>" . $row['joining\_date'] ."</td>";

$sno++;

echo "<td><form action='read\_member.php' method='post'><input type='hidden' name='name' value='" . $uid . "'/><input type='submit' class='a1-btn a1-blue' id='button1' value='View History ' class='btn btn-info'/></form><form action='edit\_member.php' method='post'><input type='hidden' name='name' value='" . $uid . "'/><input type='submit' class='a1-btn a1-green' id='button1' value='Edit' class='btn btn-warning'/></form><form action='del\_member.php' method='post' onsubmit='return ConfirmDelete()'><input type='hidden' name='name' value='" . $uid . "'/><input type='submit' value='Delete' width='20px' id='button1' class='a1-btn a1-orange'/></form></td></tr>";

$msgid = 0;

}

}

}

}

?>

</tbody>

</table>

<script>

function ConfirmDelete(name){

var r = confirm("Are you sure! You want to Delete this User?");

if (r == true) {

return true;

} else {

return false;

}

}

</script>

<?php include('footer.php'); ?>

</div>

</body>

</html>

**5.2 Code efficiency**

Code efficiency has been achieved through proper validation using various methods in PHP coding. Fist no data can be added, viewed, edited and deleted to database without login or session. For this we have implanted session tracking techniques through PHP. Codlings have been used to validate various forms to ensure correct data should enter in database.

**5.3Testing Approach**

**TESTING PROCEDURES**

* **Unit Testing:** A Unit corresponds to a form/class in the package. Unit testing focuses on verification of the corresponding form or class. In this level we have tested all our forms/classes individually. This testing includes testing of control paths, interfaces, local data structures, logical decisions, boundary conditions, and error handling. From this testing we were able to save, retrieve, update, delete and the search records on a table.
* **Integration Testing**: Integration testing is used to verify the combination of the software modules. In this level, we have tested by combining all unit tested forms into a subsystem. Here we found that the subsystems are performing well.
* **System Testing:** System testing is used to verify, whether the developed system meets the requirements.
* **Acceptance Testing:** Acceptance is the part of the project by which the customer accepts the product. The system under consideration is tested for user acceptance by constantly keeping in touch with the system users at time of developing and making changes whenever required.

We hope that after the acceptance testing the system will perform the best result for the organization. When modification will be made, we will use regression testing during the maintenance of the system.

The Software System delivered to the customer may undergo changes. Changes may be due to addition of new functional modules or performance enhancement .For this purpose proper maintenance of the system is must.

5.3.1 Unit Testing

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case Ref No** | | **TCT-001** | | |
|  | |  |  | |
| Functionality | | : | Log in to the System | |
|  | |  |  | |
| Expected outcome | | : | The user should not login to unspecified area  and some error message follow | |
|  | | | | |
| Step No. | Data Used | | | Actual Outcome |
| 1. | Click on the log in button  without entering username or password | | | An alert message came to enter  username |
| 2. | Click on the log in button  after entering some username leaving password field blank | | | An alert message came to enter  password |
| 3. | Click on the log in button  after entering some password but leaving username field blank | | | An alert message came to enter  username |
| 4. | Click on the log in button  after entering some wrong username but correct password | | | A message displayed on Log in  page about this |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case Ref No** | | **TCT-002** | | |
|  | |  |  | |
| Functionality | | : | Enter valid Data for member registration | |
|  | |  |  | |
| Expected outcome | | : | The user should not get register any record without filling all necessary fields and some error message follow  The user should not get registered again with same member id | |
|  | | | | |
| Step No. | Data Used | | | Actual Outcome |
| 1. | Click on the submit button  without entering valid details | | | An alert message came to each  details and focused on the respective fields |
| 2. | Click on the reset button  After entering member detail. | | | All the field that are manually entered will become blank. |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case Ref No** | | **TCT-003** | | |
|  | |  |  | |
| Functionality | | : | Add Payment of particular member | |
|  | |  |  | |
| Expected outcome | | : | The user should not add payment of any one member’s  payment in another’s payment record. Without filling all necessary fields and some error message will be followed. | |
|  | | | | |
| Step No. | Data Used | | | Actual Outcome |
| 1. | Click on that plan available from the list that is chosen by the member for the payment. | | | All the detail of the plan student will be displayed and get focused on the respective fields |
| 2. | Click on the cancel button  After entering member detail. | | | All the field that are manually entered will become blank and will return to the member detail page. |

**6. RESULTS AND DISCUSSION**

**6.1 Test Reports**

|  |  |  |
| --- | --- | --- |
| Test Case No | Date | Pass / Fail |
| TCT-001 | 11/5/2018 | Pass |
| TCT-002 | 02/6/2018 | Pass |
| TCT-003 | 19/6/2018 | Pass |

**7. DOCUMENTATION**

* + **For Management**

Gym Management System is primarily designed for providing information from the data after processing them. This system is designed for supplying information to the strategic level of management from the operational control. It includes almost all the functional areas needed like keeping Member Record and payment Record.

* + **For User**

With this electronic data processing system, the operators will able to maintain the following task:

* + - Information regarding Members.
    - Records profile detail and payment detail.
    - Regular Transaction Details
  + **For data processing department**
* In maintenance, the data processing department needs to create backup of the database file from time to time.
* The main menu of the system provides different menus for different purposes.

**FUTURE APPLICATION**

Software development is never –ending process and continues the life of the software as per the changing needs of the user from time to time. The project is no doubt has been developed keeping in mind easy modification and enhancement that may be required from time to time.

However, there are many scopes to modify this software. As because due to shortage of time, we here become unable to include many things. We are trying to cover all their existing system for keeping records of the members enrolls but due to shortage of time we become unable to include many things. Due to lake of time I here include none of them and a future scope one can develop these returns which are so much essential. Only with a little more doing it is possible to design the formats for those returns. Moreover, an on-line system will be more helpful to the organization. With almost the same data with only a little modification an on-line system can be designed to fulfill their demands. All these can be considered to be future scope for this project.

**CONCLUSION**

After implementing the application it will contain the advantages were incomparable to the present contemporary systems used by company. The most admirable feature founded was its simplicity in terms of application to the user but its highly beneficial outputs can’t be ignored. The users will be highly benefited after using the system.

It is hoped that this project will help the future developers to modify and implement the system. After modifying some techniques of the programs, it will give us the best performance as our requirements. The project will be very useful for the users.